B-014

ハイブリッドクラウド活用に向けたアプリケーションの性能とコストの最適化に関する検討
Research on Optimizing Application Performance and Cost in Hybrid Clouds

礒田 有哉† 坂田 匡通† 石井 陽介† 谷川 桂子†
Yuya Isoda Masayuki Sakata Yousuke Ishii Keiko Tanigawa

1. はじめに

近年，パブリッククラウドの活用が普及しているが，ミッションクリティカルやビッグデータ，個人情報や機密情報を扱うサービスではオンプレミスを活用する機会も依然多い。また，事業継続性やベンダーロックイン回避，コスト最適化にも起因し，マルチクラウドを含むハイブリッドクラウドでシステムを構築するニーズも高まっており，アプリケーションをハイブリッドクラウドで運用する技術が求められている。

アプリケーションをハイブリッドクラウドで運用するとき，アプリケーションをプラットフォームごとに構築する技術や最適化する技術が求められる。前者は，VM やコンテナ，kubernetes（以下，k8s と省略する）などの仮想化技術により，プラットフォームに依存することなくアプリケーションを利用することが可能になった。一方で，後者は，プラットフォームに閉じた研究開発は進んでいるが，ハイブリッドクラウドで汎用的に利用可能な技術は確立されていない。

この背景から，我々は，ハイブリッドクラウドでアプリケーションの性能と費用を持続的に最適化する技術について研究を推進している。

2. 課題

近年，DevOps ツールの進化に伴い，アプリケーションやプラットフォームを高頻度に更新することが可能になった。我々は，アプリケーションの性能はプラットフォームと Pod サイズ（CPU や Memory），Pod 数に応じて定まると仮定し，ある要求性能を満たすアプリケーションを提供する際に，プラットフォームごとに必要な Pod サイズや Pod 数を見積もることで，システム全体の費用削減が可能であると考え，研究を推進している。

まず，上記仮説と費用の削減効果を確認するために予備実験を実施した。予備実験では，scikit-learn real world datasets で公開されているニュース記事の分類処理をアプリケーションに用い，異なるプラットフォーム，異なる Pod サイズ（CPU）におけるアプリケーションの性能を確認する。本実験では，CPU の世代変化に伴う性能と費用を確認するために，表 1 に記載のプラットフォーム（AWS EC2 インスタン）を用いて実験した。図 1 の実験結果は，X 軸がアプリケーションに割り当てた Pod サイズ（CPU），Y 軸がアプリケーション性能（1 秒あたり処理したニュース記事数）であり，試行回数 10 回の平均スループットを記載している。

この実験結果から，プラットフォームごとにアプリケーションの要求性能を達成するために必要な Pod サイズを最適化することで，システム全体の費用を削減可能であることを確認できた。例えば，要求性能が 400 [pages/sec] の場合，m5.xlarge では 2.0 CPU が必要だが，c6i.xlarge では 1.4 CPU で要求性能を達成できる。即ち，m5.xlarge から c6i.xlarge にプラットフォームを移行することで，CPU を 30.0%削減でき，費用を 39.6%削減できる見通しを得た。

表 1　予備実験環境

	AWS m4.xlarge	AWS m5.xlarge	AWS c6i.xlarge
vCPU	4	4	4
CPU MHz	2300.05	3103.23	3513.27
Memory	16 GiB	16 GiB	8 GiB
Cost [USD/h]	0.258	0.248	0.214
kubernetes	1.20.7	1.20.7	1.20.7

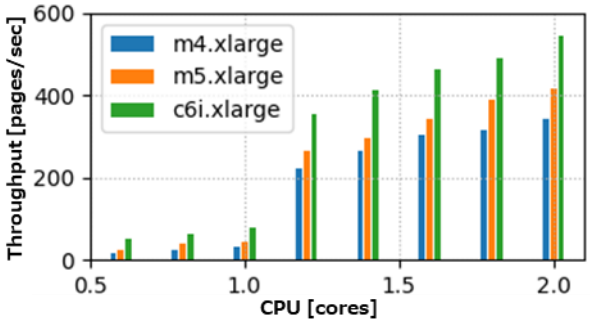

図 1　予備実験結果

既存技術として，Kubernetes では，Pod のオートスケーリング技術を実装しており，Pod サイズの自動調整を Vertical Pod Autoscaler（VPA），Pod 数の自動調整を Horizontal Pod Autoscaling（HPA）で提供する[1,2]。これらは，k8s のシングルクラスタや同種のプラットフォームにおけるマルチクラスタでは有用な機能であるが，本研究で対象にするハイブリッドクラウドではプラットフォームごとにアプリケーションの性能傾向が異なることを前提にしており利用することができない。具体的には，VPA がクラスタ A で学習した結果をクラスタ B にそのまま流用することができない。

これらのことから，ハイブリッドクラウドでアプリケーションの性能と費用を持続的に最適化するためには，プラットフォームごとの特性を考慮した VPA や HPA の検討が必要となる。

また，アプリケーションを実行可能な k8s クラスタを特定するために，クラスタやノードごとに稼働状況を監視・予測することも必要となる。既存ノードに対してアプリケーションをデプロイする場合には，追加の費用は発生しないが，新規ノードを追加する場合には，追加の費用が発生する。このため，既存ノードを積極的に活用する方法が第一の選択肢となり，新規ノードを追加する方法は第二の選択肢となる。

† 株式会社 日立製作所，Hitachi, Ltd.

本稿では，これらのことを考慮し，ハイブリッドクラウドでアプリケーションの性能と費用を持続的に最適化する方法について検討した。

3. 提案

本研究では，k8s を用いたハイブリッドクラウドにおいて，k8s クラスタごとにシステム構成（CPU，Memory，Disk など）が異なる際に，アプリケーションのリソース（Pod 数や Pod サイズ）を動的に設定することで，k8s クラスタによらず同じアプリケーション性能を提供するサービスの確立を目指している。本稿では，本サービスを提供するための設計と機能について提案する。

まず，アプリケーションの性能を定義する。一般に，アプリケーションの性能は，レイテンシやスループットなどで表現されることが多いが，多種多様な命令を受け付けるアプリケーションでは，性能を定義することは困難である。例えば，データベースでは，簡単～複雑な検索命令を受け付けられるため，命令ごとにレイテンシは大きく異なる。このため，アプリケーションの性能は，ある性能試験に対する性能と定義する必要がある。この関係性を図 2 に記載する。即ち，本サービスでは，アプリケーションと性能試験，目標性能をユーザが入力することを想定している。

図 2　アプリケーションと目標性能の関係性

次に，ユーザの指示に基づく，アプリケーションの性能と費用の最適化について提案する。最適化処理は，①初回デプロイ，②クラスタ内最適化，③クラスタ間最適化の 3 段階に分類でき。処理の概要を図 3 に記載する。

初回デプロイでは，利用可能な k8s クラスタや k8s ノードを CPU や Memory などのリソース予測に基づいて導出し，利用可能な k8s ノードごとにアプリケーションのリソース（Pod 数や Pod サイズ）を導出し，アプリケーションのリソースを上回る余剰リソースを持つ k8s ノードを抽出する。そして，抽出した k8s ノードごとにアプリケーションの利用費用を算出し，費用が最小となる k8s ノードにアプリケーションをデプロイする。また，デプロイ後，アプリケーションの性能がユーザの目標性能を満たしているか確認するために，性能試験を非定期／定期的に実施する。このとき，得られた性能結果は，アプリケーションのリソース推論の精度向上のために活用する。

次に，デプロイ後，ユーザの目標性能が達成たされないことを検出／予測すると，VPA や HPA を用いたアプリケーションのリソース最適化を実行する。このとき，VPA や HPA の実行に伴う k8s クラスタや k8s ノードのリソース不足を検出／予測すると，k8s クラスタ間でアプリケーションをマイグレーションする必要がある。その際は，初回デプロイと同様に，費用が最小となる k8s ノードを導出し，この k8s ノードにアプリケーションをマイグレーションする。

このように，ハイブリッドクラウドでアプリケーションを運用することで，ユーザの目標性能を達成しつつ，最小費用でアプリケーションを提供できると考えている。

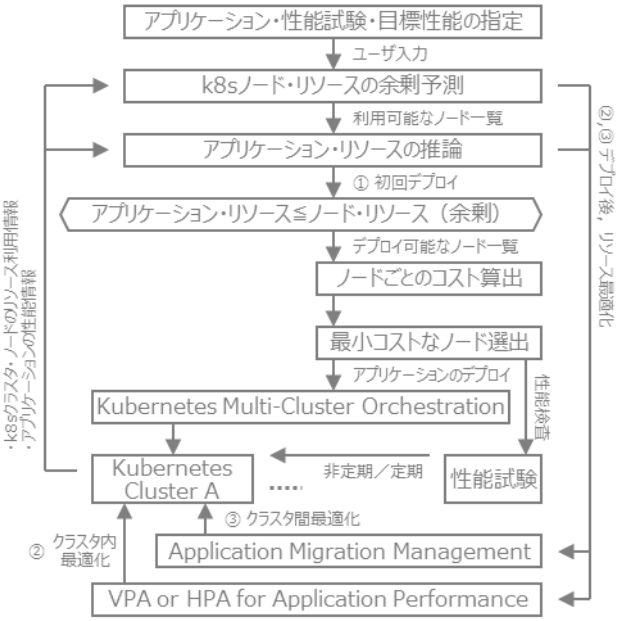

図 3　アプリケーションの性能と費用の最適化処理概要

4. おわりに

本稿では，k8s を用いたハイブリッドクラウドにおいて，k8s クラスタごとにシステム構成が異なる際に，プラットフォームごとにユーザの要求性能に応じたアプリケーションのリソースを推測することで，アプリケーションの性能を維持しつつシステム全体の費用を削減するサービスの設計と機能について検討した。また，本研究の課題と解決した際に得られる効果について明らかにした。

今後，本サービスの実現に向けて，課題と効果の詳細化，機能開発および効果検証を実施する予定である。課題と効果の詳細化では，パブリッククラウドやオンプレミス，アプリケーション，運用シナリオなどを追加し，最も効果があるシナリオと本サービスの提供範囲を評価実験により確認する。また，機能開発では，k8s 稼働状況の予測，プラットフォームごとのアプリケーションリソースの予測，アプリケーションのデプロイ制御，アプリケーション性能の収集管理，アプリケーション性能に基づく VPA や HPA などの検討・開発が必要となる。この中で最も重要視している機能は，アプリケーションのリソース予測とデプロイ制御である。リソース予測では，無限の時間と予算があれば，全てのプラットフォームでアプリケーションを実行することで，プラットフォームごとに必要なリソースを予測できる。この時間と予算を削減することが課題となる。デプロイ制御では，アプリケーションが利用可能なプラットフォームの選択肢を拡充し，費用削減効果を高める技術開発が必要となる。アプリケーションは，プラットフォーム（環境，データ，その他サービスなど）に依存することも多いため，依存性を動的に検出・排除する技術が課題となる。

参考文献

[1] Kubernetes Autoscaler, https://github.com/kubernetes/autoscaler, 2022/3.
[2] Horizontal Pod Autoscaling, https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale, 2022/3.